(11) Japanese Patent Application Unexamined Publication No. 3-2964

(43) Publication Date: January 9, 1991

(21) Application No. 1-135822

(22) Application Date: May 31, 1989

(72) Inventor: Toru YAMAKITA

(71) Applicant: CASIO COMPUTER Co., Ltd.

(74) Agent: Patent Attorney, Toshimasa MACHIDA

SPECIFICATION

1. Title of the Invention: DOCUMENT DATA PROCESSING APPARATUS

2. Claim

A document data processing apparatus comprising:

external storage means for storing document data with a file name associated with the document data,

document data storage means for storing the document data,

file name generating means for generating the file name by updating a predetermined character in a predetermined sequence order,

detecting means for detecting that the file name generated by the file name generating means is stored in the

external storage means and generating a next file name by commanding the file name generating means to update the character if the detecting means detects that the file name generated by the file name generating means is stored in the external storage means, and

writing means for writing the file name together with the document data in the document data storage means onto the external storage means if the detecting means detects that the file name generated by the file name generating means is not stored in the external storage means.

3. Detailed Description of the Invention

[Technical Field of the Invention]

The present invention is related to a document data processing apparatus such as a wordprocessor.

[Description of the Related Art]

In known wordprocessors, a user reads document data stored in a disk, adds data to the read data or corrects the read data, and then transfers the modified data as another document file to the disk for storage. Each time the user needs to enter a file name.

[Problem to be Solved by the Invention]

Inputting a file name increases workload on the user.

The user also needs to create a file name that allows the user to easily recall that the file is closely related to the document prior to addition and correction. This inconveniences the user, and storage of the data to the disk cannot be performed quickly. If the same file name is already used, another file name needs to be entered again.

This inconvenience is attributed to the fact that the assignment of an appropriate file name subsequent to data addition or data correction is performed only by a user input operation.

An object of the present invention is to automatically generate a file name appropriate for a document to be stored.

[Means for Solving the Problems]

The present invention includes means described below.

External storage means 1 (see a functional block diagram of Fig. 1 for the external storage means 1 and other means to be discussed below) is a disk or the like for storing document data with a file name correspondingly associated therewith.

Document storage means 2 is a RAM or the like, as a text memory, for storing the document data, and stores document data subsequent to data addition or data correction performed thereto.

File name generating means 3 generates a file name by

updating predetermined characters in a predetermined sequence order. The predetermined characters may include characters "A, B, C, D, ..., Z, A, B, ..." or "A, I, U, E, O, ... NN, A, I, ...", numerals "1, 2, 3, ..., 10, ...", "A1, A2, ...", "First, Second, ...", symbols, or codes.

Detecting means 4 detects whether the file name generated by the file name generating means 3 is stored in the external storage means 1, and commands the file name generating means 3 to update the character to generate a next file name if the file name generated by the file name generating means 3 is stored in the external storage means 1.

Writing means 5 writes the file name together with the document data in the document storage means 2 onto the external storage means 1 if the detecting means 4 detects that the file name generated by the file name generating means 3 is not stored in the external storage means 1.

[Operation]

The operation of the means of the present invention is described below.

The document data read from the external storage means 1 is subjected to data correction, data addition, etc., and the resulting document is then transferred as another document file to the external storage means 1 for storage.

In response to an operation of a predetermined document

storage key, the file name generating means 3 generates the file name by updating a predetermined character in a predetermined sequence order. For example, the file name prior to the data correction or data addition is "conference minutes A", the document storage means 2 updates the character "A" to "B", thereby generating the file name "conference minutes B".

The detecting means 4 detects whether the generated file is stored in the external storage means 1. If the detecting means 4 detects that the same file name is stored as another file name in the external storage means 1, the detecting means 4 commands the file name generating means 3 to update the character to generate another file name, for example, "conference minutes C". If the detecting means 4 detects that the same file is not stored as another file name in the external storage means 1, the writing means 5 writes the generated file name together with the document data in the document storage means 2 onto the external storage means 1.

A file name optimum for the document to be stored is thus automatically generated.

[Embodiments]

One embodiment of the present invention is described below with reference to Figs. 2-4.

Fig. 2 is a block diagram illustrating a basic structure of a wordprocessor.

CPU 11 receives document data input from an input unit 12, and causes a display 13 to display on a text screen thereof. The CPU 11 converts the document data into a Kanji-mixed sentence by searching a dictionary memory 14-1 in a ROM 14, and stores the converted data in a document memory 15-1 in a RAM 15.

A keyboard of the input unit 12 includes character keys, an execution key, a next candidate key, a new document storage key KA, and a document updating and storage key KB. The new document storage key KA is used to issue a command to transfer document data in the document memory 15-1 to a floppy disk drive (FDD) 16. In response to an operation of the new document storage key KA, the CPU 11 updates a predetermined character that is to be updated in accordance with a predetermined sequence order, thereby generating a file name. On condition that the generated file name is not stored as a file name of another document in the FDD 16, the CPU 11 transfers, to an FDD controller 17, the generated file name together with the document data in the document memory 15-1 to be written onto the FDD 16. Work memories WM1, WM3U, and WM3S in the RAM 15 are used in this case. The work memory WM1 stores a file name read from an index area of the FDD 16. The work memory WM3U stores a user file

name a user manually inputs. The work memory WM3S stores a system file name that has been produced by automatically updating the predetermined character in accordance with the predetermined sequence order. A file name containing the user file name and the system file name becomes a file name of a newly produced document. For example, "conference minutes" of the "conference minutes A" is a user file name, and the character "A" of the "conference minutes A" is a system file name. The document updating and storage key KB is used to issue a command to transfer the updated document data instead of original document data to the FDD 16 when the original document data is read and updated.

The ROM 14 includes a character generator CG in addition to the dictionary memory 14-1. When the document data is output from within the document memory 15-1, the character generator CG converts the document data into a dot pattern to output the dot pattern to a display 13 as a CRT display or to output the dot pattern to the printer unit 18 as a thermal printer.

Fig. 3 illustrates the structure of the FDD 16. The FDD 16 includes an index area IA and a document area DA. Stored in the index area are a file name corresponding to document data of various types stored in the document area DA, and a pointer value pointing to a storage location of stored document.

The operation of this embodiment is described below with reference to a flowchart of Fig. 4.

The flowchart of Fig. 4 starting with step S1 represents the operation that is performed in response to the operation of the new document storage key KA. The new document storage key KA is used to store, onto the FDD 16, document data that is newly produced and input. When original document having a designated file name is read from the FDD 16, and subjected to data correction, data addition, etc., the new document storage key KA is also used to store, onto the FDD 16, the resulting document data as a document separate from the original document data.

When the new document storage key KA is operated, the CPU 11 determines whether a user file name is stored in the work memory WM3U (step S1). The work memories WM3U and WM3S (hereinafter collectively referred to as a work memory WM3) store the user file name and the system file name forming the file name when the document data of the designated file name is read from the FDD 16. If document data is newly produced and input, the work memory WM3 does not store the file name. If it is detected in step S1 that no user file name is stored in the work memory WM3U, in other words, if the document data is newly produced, a user file name is input and set in the work memory WM3U (step S2).

The CPU 11 reads all file names from the index area IA

in the FDD 16, and writes the file names onto the work memory WM1 (step S3).

The CPU 11 checks whether the system file name is stored in the work memory WM3S (step S4), in other words, whether the document data is newly created, the CPU 11 proceeds to step S10. The CPU 11 generates a system file name and enters initial settings onto the work memory WM3U. For example, if the system file name is "A, B, C, ..., Z, A, B, ...", the system file name to be initially set in the work memory WM3U becomes "A". If the system file name is set in the work memory WM3S, in other words, the document data having the designated file name is read from the FDD 16, CPU 11 proceeds to step S5. The CPU 11 performs an increment process to update the system file name in the work memory WM3S in accordance with the predetermined sequence order. Here, alphabets "A", ..., "Z" correspond to JIS codes "0333", ..., "0358", respectively. To update the character from "A" to "B", "1" is simply added to a "code of A (0333)". Number "0", ..., "9" respectively correspond to JIS codes "0316", ..., "0325". In a similar manner as above, the system file name is updated.

If the initial setting or the updating process is performed on the system file name, the file name composed of the user file name and the system file name is generated in the work memory WM3. In step S6, the CPU 11 determines

whether the file name generated in the work memory WM3 is stored in the work memory WM1. If it is determined that the generated file name is stored in the FDD 16, the CPU 11 returns to step S5. After updating the system file name, the CPU 11 searches the content of the work memory WM1, and repeats the updating process until a file name not present in the FDD 16 is generated.

If the file name not present in the FDD 16 is generated, the CPU 11 proceeds to step S7. The user file name and the system file name in the work memory WM3 are displayed together with a guidance message. In this case, a guidance message "storing file name XXX (content of the work memory WM3)?" is displayed, for example. The CPU 11 waits on standby for key inputting (step S8). The user checks the content of the display, and if a desired file name is displayed, the user operates the execution key. If the desired file name is not displayed, the next candidate key or the preceding candidate key is operated. If the next candidate key is operated, the CPU 11 returns to step S5 to perform the increment process on the system file name. If the preceding candidate key is operated, the CPU 11 proceeds to step S9 to perform a decrement process on the system file name. On condition that the newly generated file name is not present in the FDD 16 (step S6), the guidance message is displayed (step S7). The user simply operates one of the

next candidate key and the preceding candidate key until a desired file name is displayed.

If the execution key is operated, the CPU 11 proceeds to step S13 to write the document data in the document memory 15-1 onto the document area DA in the FDD 16. The CPU 11 writes the file name in the work memory WM3 together with the pointer value onto the index area IA in the FDD 16 (step S14).

The above-described document storage process is performed in response to the operation of the new document storage key KA. The user may read any document data from the FDD 16, update the read document data, and then store the updated document data in place of the original document data in the FDD 16. In that case, the document updating and storage key KB is used. Fig. 4 illustrates an operation performed in accordance with a flowchart starting with step S11.

In response to the operation of the document updating and storage key KB, the file name in the work memory WM3 is output and displayed together with a guidance message (step S11). When document data to be updated is read together with a file name from the FDD 16, the work memory WM3 stores the file name. In this case, a guidance message "updating a file name XXX?" is displayed. The CPU 11 proceeds to step S12 to wait on standby for key inputting. If a cancel key

is operated, the process ends. If the execution key is operated, the CPU 11 proceeds to step S13. The document data in the document memory 15-1 is written onto the document area DA in the FDD 16. The file name and the pointer value thereof are written onto the index area IA of the FDD 16 (step S14).

In the above example, the "conference minutes A", and the "conference minutes B" are used. A "first volume" and a "second volume" may be used. The system file name is not limited to a single character. A character string composed of a plurality of numbers, such as "100", "200", ..., may be used.

[Advantages]

In accordance with the present invention, a file name optimum for a document to be stored is generated. Workload on the user is reduced. Data storage to the disk is quickly performed. File name duplication is reliably prevented. A mnemonic file name helping the user to remember a relationship with the file prior to data addition or data correction is automatically generated, and file management is easily performed.

4. Brief Description of the Drawings

Fig. 1 is a functional block diagram of the present

invention. Figs. 2-4 illustrate the embodiment of the present invention. Fig. 2 is a block diagram illustrating the basic structure of a wordprocessor, Fig. 3 illustrates the structure of the FDD 16, and Fig. 4 is the flowchart illustrating the operation of the wordprocessor.

11.....CPU, 14.....ROM, 14-1.....dictionary memory, 15.....RAM, 15-1.....document memory, 16.....FDD, 17.....FDD controller

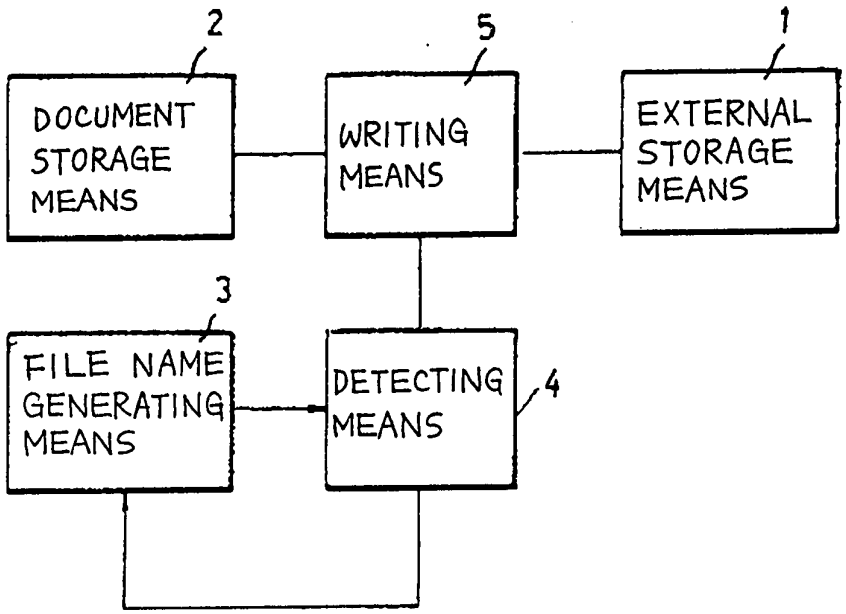
2
5
1
DOCUMENT STORAGE MEANS
WRITING MEANS
EXTERNAL STORAGE MEANS
3
FILE NAME GENERATING MEANS
DETECTING MEANS
4

FIG. 1

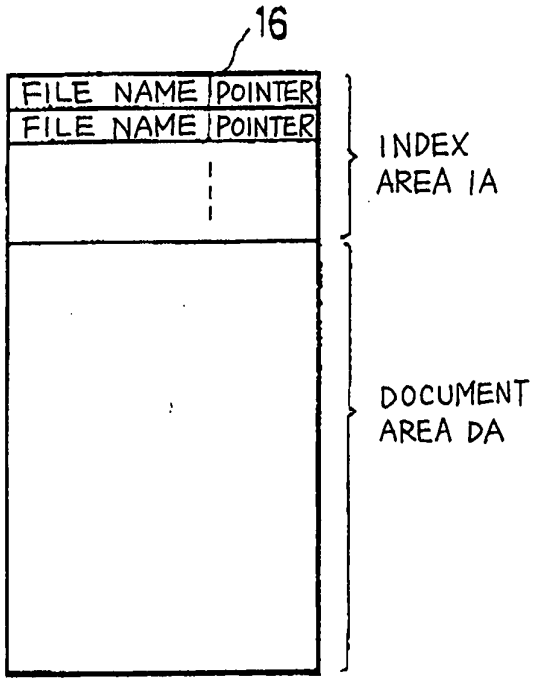
16
FILE NAME
POINTER
FILE NAME
POINTER
INDEX AREA IA
DOCUMENT AREA DA

FIG. 3

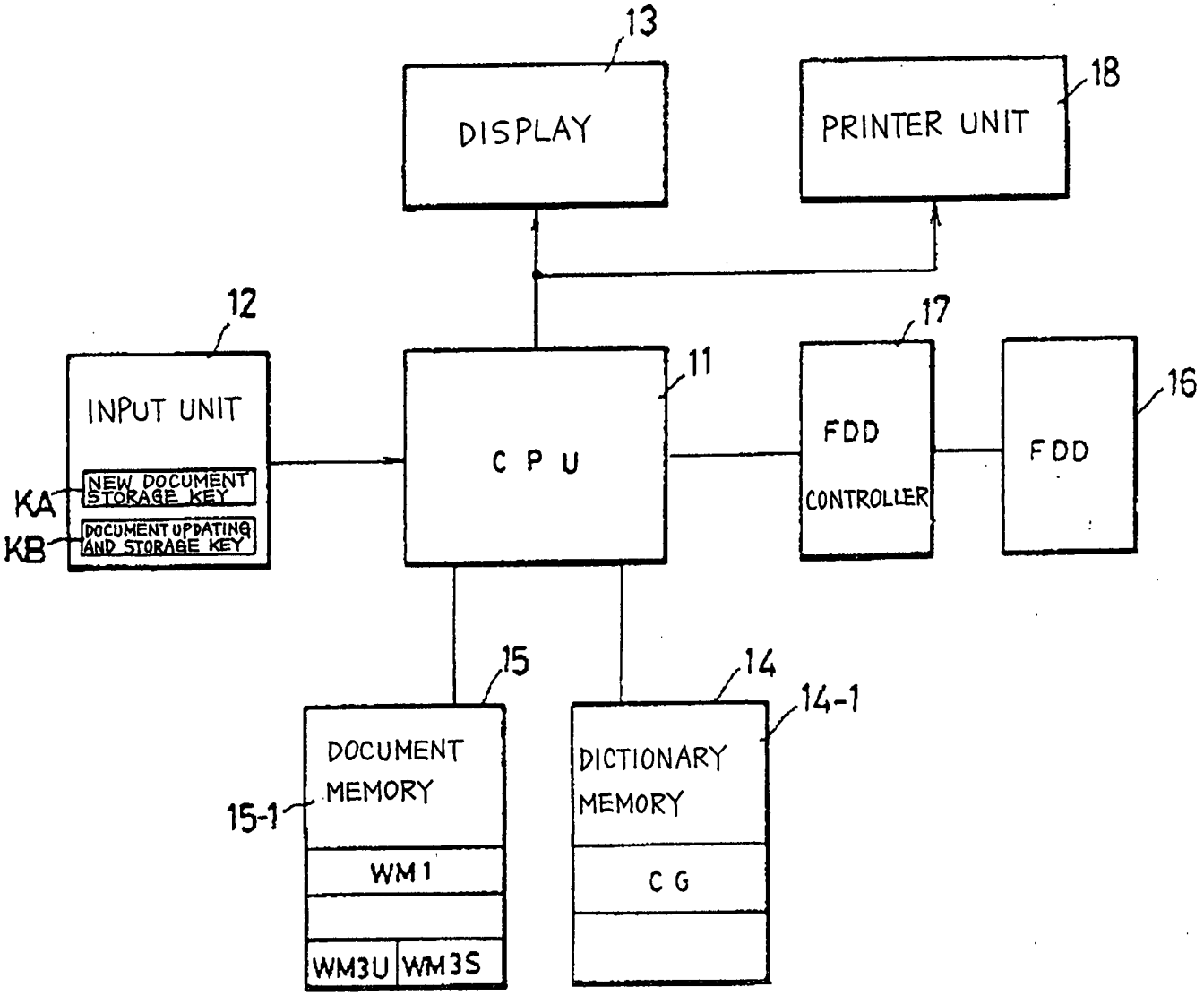
13
DISPLAY
18
PRINTER UNIT
12
INPUT UNIT
NEW DOCUMENT STORAGE KEY
KA
DOCUMENT UPDATING AND STORAGE KEY
KB
11
C P U
17
FDD CONTROLLER
16
FDD
15
15-1
DOCUMENT MEMORY
WM1
WM3U
WM3S
14
14-1
DICTIONARY MEMORY
C G

FIG. 2

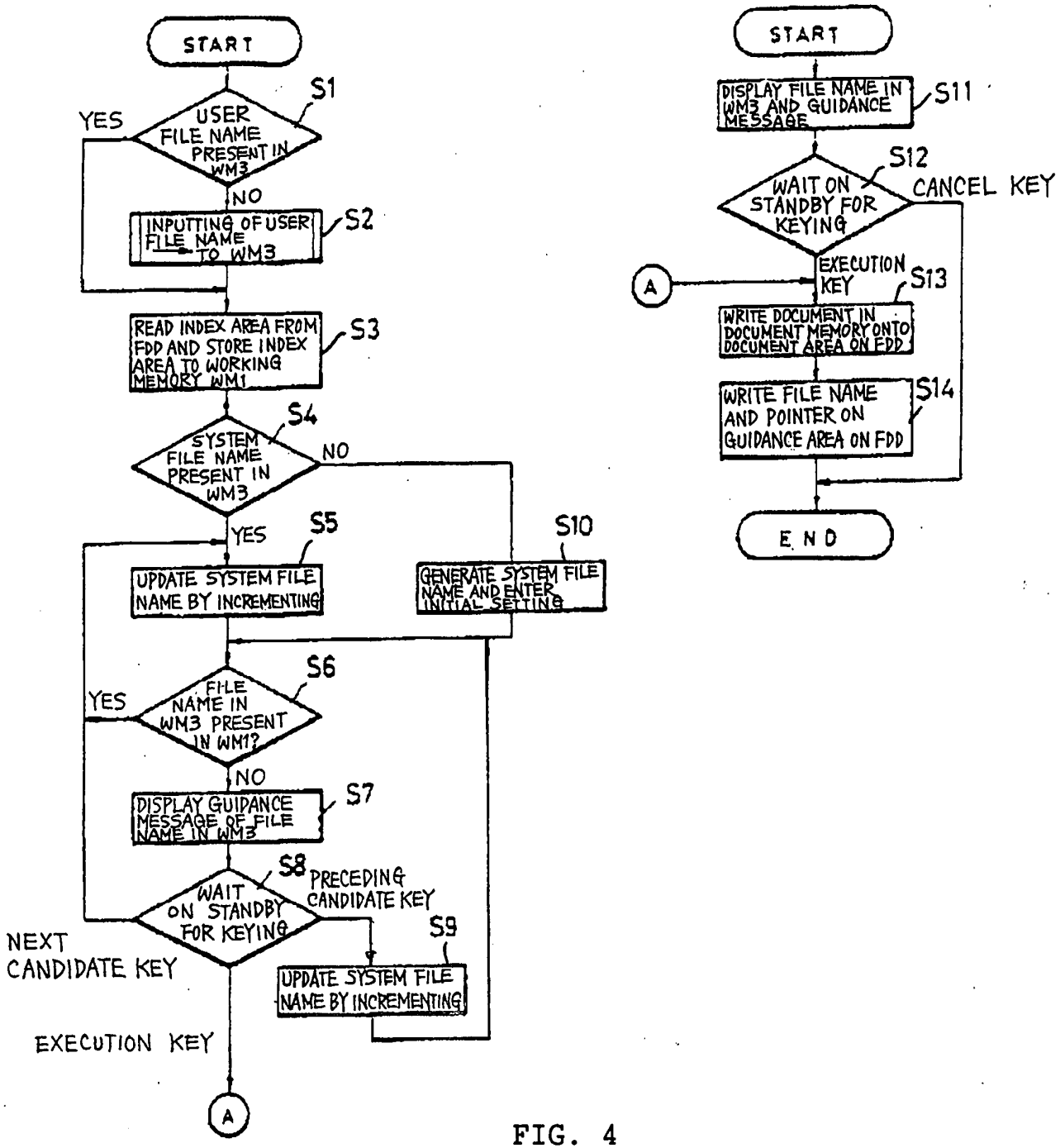
START
S1
USER FILE NAME PRESENT IN WM3
YES
NO
INPUTTING OF USER FILE NAME TO WM3
S2
READ INDEX AREA FROM FDD AND STORE INDEX AREA TO WORKING MEMORY WM1
S3
S4
SYSTEM FILE NAME PRESENT IN WM3
NO
YES
S5
UPDATE SYSTEM FILE NAME BY INCREMENTING
S10
GENERATE SYSTEM FILE NAME AND ENTER INITIAL SETTING
S6
FILE NAME IN WM3 PRESENT IN WM1?
YES
NO
DISPLAY GUIDANCE MESSAGE OF FILE NAME IN WM3
S7
S8
WAIT ON STANDBY FOR KEYING
PRECEDING CANDIDATE KEY
S9
NEXT CANDIDATE KEY
UPDATE SYSTEM FILE NAME BY INCREMENTING
EXECUTION KEY
A
START
DISPLAY FILE NAME IN WM3 AND GUIDANCE MESSAGE
S11
S12
WAIT ON STANDBY FOR KEYING
CANCEL KEY
A
EXECUTION KEY
S13
WRITE DOCUMENT IN DOCUMENT MEMORY ONTO DOCUMENT AREA ON FDD
WRITE FILE NAME AND POINTER ON GUIDANCE AREA ON FDD
S14
END

FIG. 4